An Asynchronous Decomposition Algorithm for Security Constrained Unit Commitment under Contingency Events

Kibaek Kim,* Mihai Anitescu,*† and Victor M. Zavala*‡
*Mathematics and Computer Science Division
Argonne National Laboratory, Lemont, IL, USA
†Department of Statistics
The University of Chicago, Chicago, IL, USA
‡Department of Chemical and Biological Engineering
University of Wisconsin-Madison, Madison, WI, USA

***Abstract*—We present a parallel and asynchronous decomposition algorithm for solving security constrained unit commitment (SCUC) problem under contingency events in system components in combination with uncertain wind power generation. The problem is formulated as a two-stage stochastic mixed-integer program, where the first stage schedules slow generators and the second stage adjusts the schedules for fast generators and dispatches power for a given contingency event. Our algorithm uses an asynchronous variant of the Lagrangian dual decomposition that can better handle high imbalance in computational times for scenario subproblems (which are themselves mixed-integer programs). We also present a special case of the SCUC problem that considers a deterministic wind power generation and develop a Benders decomposition for solving this special case. In our computational study, we solve the SCUC model for the WECC 225-bus system that incorporates the single contingency events on transmission lines. We show that the asynchronous algorithm can significantly reduce solution times (by up to a factor of eight) of an off-the-shelf synchronous dual decomposition algorithm.**

***Index Terms*—Asynchronous parallel computing, security-constrained unit commitment, dual decomposition**

I. Introduction

The development of security criteria is key to achieve reliability and stability of the power grid. The $n-K$ security criterion, in particular, ensures that the power system is capable of satisfying the load for any possible combinations of K out of n grid component failures. In particular, North American Electric Reliability Corporation has established transmission system planning performance requirements, which suggests the system being sustainable to every single component failure without load loss; that is, $n-1$ compliance.

This material is based upon work supported by the U.S. Department of Energy, Office of Science and Office of Electricity Delivery and Energy Reliability, under contract DE-AC02-06CH11357. We gratefully acknowledge the computing resources provided on Bebop, a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory.

Security constrained unit commitment (SCUC) model has been used for modeling unit commitment with contingency events in load forecast, generation forecast, and component failures (e.g., [1], [2]). Papavasiliou and Oren [2] formulated the SCUC model as a two-stage stochastic mixed-integer program (SMIP), where the first stage schedules slow generators and the second stage schedules fast generators and dispatches power in order to satisfy load for a given uncertain contingency realization. The authors also presented the numerical examples using a test instance with 30 scenarios (10 wind scenarios and 2 contingency events).

In this paper, we present parallel decomposition methods for SMIP problems of the SCUC model, as used in [2]. In particular, we consider single contingency events on transmission lines (i.e., single line failures) as well as uncertain wind power generation. The SMIP problems that arise in our SCUC model are computationally challenging because the size of the problem (particularly, the number of binary variables) increases in the number of scenarios and because binary variables appear in the first and second stage. This prevents the use of Benders-type methods (e.g., [3], [4]).

We develop a dual decomposition (DD) method for solving the SCUC with a number of contingency scenarios. DD is the Lagrangian relaxation of the nonanticipativity constraints [5], which provides an effective parallel decomposition approach to solve this type of SMIP (e.g., [6], [7]). A key component of the DD method is the procedure to find a sequence of dual variables for the Lagrangian dual function. In our DD method, we apply a recently-developed bundle method with trust-region (TR) constraints, which we call bundle-trust-region (BTR) method. The BTR method is a variant of bundle methods that solves a master problem for finding a new dual variable with the regularization of the search space by the TR constraints. The master problem is obtained by approximating the Lagrangian dual function with a set of linear inequalities. The key advantage of the bundle methods is the finite termination of algorithm at optimum (i.e., an optimal Lagrangian dual value in the DD method).

The subgradient method is another common approach to update dual variables, which uses a subgradient of the Lagrangian dual function with step-size rules (e.g., [2], [6]). However, finding an optimal step-size is computationally impractical in such methods. The subgradient method requires ad-hoc step-size selection criteria and thus suboptimal stopping criteria, as compared with the finite termination at optimum by the bundle methods.

Moreover, in the context of parallel DD method, another challenge is that the scenario (event) subproblems lead to severe imbalances in computing time, because different sets of component failures induce different network topologies [8]. To address this issue, we also present the asynchronous variant of the BTR method and the computational performances that show significant reduction in solution time.

We also develop a Benders decomposition method for solving the SCUC with all single contingency scenarios for a given wind power generation, where the contingency scenario subproblems are solved in parallel. A key observation is that a feasibility cut can be generated based on a binary logic when the first stage has binary variables only. Note, however, Benders-type optimality cuts cannot be generated due to the second-stage binary variables (see discussion in [7]). Our Benders method is used to identify a subset of *active* contingency scenarios that generate Benders-type feasibility cuts. In our computational study, we use the subset of active contingency scenarios in combination with wind power scenarios.

The rest of the paper is organized as follows. Section II describes the SCUC model and presents a special case of the model. In Section III we present asynchronous parallel dual decomposition methods for SCUC. Section IV presents numerical experiments by using WECC 225-bus test system. We discuss the conclusion and future work in Section V.

II. Scurity Constrained Unit Commitment

The SCUC is formulated as a two-stage SMIP, where the first stage schedules generators and the second stage reschedules fast generators and redispatches power in order to minimize the operation cost while satisfying the feasibility for any given contingency scenarios. The scenarios are defined by a combination of wind scenarios and contingency scenarios. Each contingency represents failure on a transmission line.

A. Nomenclature

We define sets, parameters, and variables that are used for our optimization models. The units are described in brackets, if needed.

Sets

$\mathcal{D}; \mathcal{D}_n$	set of loads; loads at bus n
$\mathcal{G}; \mathcal{G}_n$	set of generators; generators at bus n
$\mathcal{G}_S$	set of slow generators
$\mathcal{L}$	set of transmission lines
$\mathcal{L}_n^+$	set of transmission lines to bus n
$\mathcal{L}_n^-$	set of transmission lines from bus n
$\mathcal{N}$	set of buses
$\mathcal{R}_n$	set of non-dispatchable generators at bus n
$\mathcal{S}$	set of contingency scenarios
$\mathcal{T}$	set of time periods

Parameters

B_{ls}	susceptance at line l for scenario s
C_g	power generation cost at generator g [$/MWh]
D_{nt}	load at bus n at time t [MWh]
DT_g	minimum downtime for generator g [hour]
K_g	commitment cost of generator g [$/hour]
P_g^{max}	maximum generation capacity [MWh]
P_g^{min}	minimum generation requirement [MWh]
R_g^{dn}	minimum ramp-down for generator g [MWh]
R_g^{up}	minimum ramp-up for generator g [MWh]
M_{gts}	non-dispachable power generation at generator g at time t for scenario s [MWh]
UT_g	minimum uptime for generator g [hour]
ρ	value of load loss [$/MWh]
π_s	probability of scenario s

Variables

f_{lts}	power flow in line l at time t for scenario s [MWh]
m_{gts}	power spillage at non-dispatchable generator g at time t for scenario s [MWh]
p_{gts}	power generation at generator g at time t for scenario s [MWh]
u_{gts}	commitment at generator g at time t for scenario s
v_{gts}	startup at generator g at time t for scenario s
z_l	1 if failure in line l; 0 otherwise
ϵ_s	fraction of the system load that allows shed
θ_{nts}	voltage angle at bus n at time t for scenario s

B. Formulation

We present the two-stage SMIP formulation of SCUC model that uses the contingency scenarios as well as uncertain wind power generation. We consider contingency scenarios to represent single transmission line failures. While adopting the SCUC formulation used in [2], we introduce slack variable ϵ_s in the first stage for precluding infeasible models due to forecast data. In particular, using our slack variable in combination with a large penalty cost ρ enforces that at least $(1 - \epsilon_s)$ fraction of the total system load must be served, as used for an n–K–ϵ reliability criterion in [4].

Our SCUC model is formulated by

$$\min \quad \sum_{s \in \mathcal{S}} \pi_s \left[\sum_{g \in \mathcal{G}} \sum_{t \in \mathcal{T}} (K_g u_{gts} + S_g v_{gts} + C_g p_{gts}) + \rho \epsilon_s \right] \tag{1a}$$

$$\text{s.t.} \quad u_{gts} = u_{gt,s-1}, \ \forall g \in \mathcal{G}_S, \ t \in \mathcal{T}, \ s \in \mathcal{S}, \tag{1b}$$

$$\epsilon_s = \epsilon_{s-1}, \forall s \in \mathcal{S}, \tag{1c}$$

$$v_{gts} \geq u_{gts} - u_{g,t-1,s} \ \forall g \in \mathcal{G}, \ t \in \mathcal{T}, \ s \in \mathcal{S}, \tag{1d}$$

$$\sum_{q=t-UT_g+1}^{t} v_{gqs} \leq u_{gts} \quad \forall g \in \mathcal{G}, \ t \in \{UT_g, \dots, T\}, \ s \in \mathcal{S}, \tag{1e}$$

$$\sum_{q=t+1}^{t+DT_g} v_{gqs} \le u_{gts}$$
$$\forall g \in \mathcal{G},\ t \in \{1,\dots,T-DT_g\},\ s \in \mathcal{S}, \tag{1f}$$
$$\sum_{l\in\mathcal{L}_n^+} f_{lts} - \sum_{l\in\mathcal{L}_n^-} f_{lts} + \sum_{g\in\mathcal{G}_n} p_{gts}$$
$$= \sum_{j\in\mathcal{D}_n} (D_{jt} - d_{jts}) - \sum_{g\in\mathcal{R}_n} (M_{gts} - m_{gts}),$$
$$\forall n \in \mathcal{N},\ t \in \mathcal{T},\ s \in \mathcal{S}, \tag{1g}$$
$$\sum_{j\in\mathcal{D}}\sum_{t\in\mathcal{T}} d_{jts} \le \epsilon_s \sum_{j\in\mathcal{D}}\sum_{t\in\mathcal{T}} D_{jts},\ \forall s \in \mathcal{S}, \tag{1h}$$
$$f_{lts} = B_{ls}(\theta_{mts} - \theta_{nts})$$
$$\forall l = (m,n) \in \mathcal{L},\ t \in \mathcal{T},\ s \in \mathcal{S}, \tag{1i}$$
$$P_g^{min} u_{gts} \le p_{gts} \le P_g^{max} u_{gts},\ \forall g \in \mathcal{G},\ t \in \mathcal{T},\ s \in \mathcal{S}, \tag{1j}$$
$$-R_g^{dn} \le p_{gts} - p_{g,t-1,s} \le R_g^{up},$$
$$\forall g \in \mathcal{G},\ t \in \mathcal{T},\ s \in \mathcal{S}, \tag{1k}$$
$$0 \le d_{jts} \le D_{jt},\ \forall j \in \mathcal{D},\ t \in \mathcal{T},\ s \in \mathcal{S}, \tag{1l}$$
$$-F_l \le f_{lts} \le F_l\ \forall l \in \mathcal{L},\ t \in \mathcal{T}, s \in \mathcal{S}, \tag{1m}$$
$$0 \le m_{gts} \le M_{gts}\ \forall n \in \mathcal{N}, g \in \mathcal{R}_n,\ t \in \mathcal{T}, s \in \mathcal{S}, \tag{1n}$$
$$0 \le v_{gts} \le 1,\ \forall g \in \mathcal{G},\ t \in \mathcal{T},\ s \in \mathcal{S}, \tag{1o}$$
$$0 \le \epsilon_s \le 1,\ \forall s \in \mathcal{S}, \tag{1p}$$
$$-180 \le \theta_{nts} \le 180\ \forall n \in \mathcal{N},\ t \in \mathcal{T},\ s \in \mathcal{S}, \tag{1q}$$
$$u_{gts} \in \{0,1\},\ \forall g \in \mathcal{G},\ t \in \mathcal{T},\ s \in \mathcal{S}. \tag{1r}$$

The objective function (1a) is the expected operation cost that includes generator startup and commitment cost, power generation cost, and penalty cost for load-shedding allowance. We assume that $\pi_s = 0$ for any contingency scenarios and $\pi_s > 0$ for the wind generation scenarios, as in [2]. Note that the constraints for each contingency scenario are still enforced to the model. We also assume that non-dispatchable power spillage m_{gts} is not penalized. The first-stage variables are u_{gts} for $g \in \mathcal{G}_S,\ t \in \mathcal{T},\ s \in \mathcal{S}$ and ϵ_s for $s \in \mathcal{S}$, representing the commitment decisions of the slow generators and the load-shedding allowance. Hence, these variables are enforced to have the same values for all the scenarios by nonanticipativity constraints (1b) and (1c). Note that the nonanticipativity constraints are not imposed on variable v_{gts} because the values of v_{gts} are given for fixed u_{gts}. This allows to have fewer first-stage variables and constraints to relax in the Lagrangian relaxation [6]. In addition to the startup decision variables v_{gts} for slow generators, the second-stage variables include the commitment and startup binary decision variables for fast generators (i.e., $\mathcal{G}\backslash\mathcal{G}_S$) and the optimal power flow variables for each scenario s. Therefore, the problem (1) has mixed-binary variables in both the first and second stages.

Constraint (1d) represents the logic between the commitment and startup decision variables. Constraints (1e) and (1f) enforce the minimum uptime and downtime, respectively, for each generator. Equation (1g) represents the network topology and enforces the balance between the power supply and demand after any load shedding and power spillage in the system. Constraint (1h) allows the pre-defined amount of total system load lost. Constraint (1i) represents the (linearized) power flow equation for each contingency scenario, where line susceptance is defined as $B_{ls} := 0$ if line l is tripped for scenario s. Constraint (1j) enforces the minimum and maximum generation capacity. Equation (1k) represents the ramping constraint. Bound constraints (1l)–(1q) are defined for each variable.

Problem (1) is challenging to solve because of the large number of binary variables $|\mathcal{G}| \cdot |\mathcal{T}| \cdot |\mathcal{S}|$ in combination with continuous variables. Moreover, the binary variables appear in both the first and second stages. One can formulate the problem (1) with respect to the first-stage variables only, where the recourse function is introduced to represent the second-stage problem parameterized by the first-stage variable value. Note, however, that the recourse function is non-convex in the first-stage variable because of the binary variables in the second stage.

C. Deterministic Wind Power Generation

We present a special case of the SCUC model (1), which considers a number of contingency scenarios for a given wind power generation (i.e., $M_{gts} = M_{gt,s-1}$ for $s \in \mathcal{S}$). We also assume that the model fixes $\epsilon_s = 0$ for $s \in \mathcal{S}$. The resulting SMIP problem has pure-binary variables in the first stage and can be simplified as follows

$$\min_{x,y_s} \quad c^T x + d^T y_0 \tag{2a}$$
$$\text{s.t.} \quad (x, y_0) \in X_0, \tag{2b}$$
$$(x, y_s) \in X_s,\ \forall s \in \mathcal{S}^C := \mathcal{S}\backslash\{0\}, \tag{2c}$$

where x is the first-stage binary variable and y_s are the second-stage mixed-binary variables for each scenario $s \in \mathcal{S}$. In particular, scenario 0 represents the non-contingency state and the other scenarios represent the contingency states. The objective function (2a) is obtained by the assumption that $\pi_0 = 1$ and $\pi_s = 0$ for $s \in \mathcal{S}^C$. Throughout the section, we assume that the problem (2) is feasible and bounded.

We observe that the SCUC model with a deterministic wind power generation is significantly easier than the stochastic counterpart and can be solved by a Benders decomposition method. We develop a Benders decomposition method for solving the non-contingency scenario problem with a number of contingency scenario subproblems for checking feasibility. Note, however, that this is not true if we have more than one non-contingency scenarios (i.e., more than one wind scenario), in which case the method requires to generate optimality cuts.

The feasibility cut can be generated to exclude a binary variable value. Suppose that $(\hat{x}, \hat{y}_0) \in X_0$ and that there is no y_s such that $(\hat{x}, y_s) \in X_s$ for some $s \in \mathcal{S}^C$. We can exclude $(\hat{x}, \cdot)$ from X_0 by adding feasibility cut

$$\sum_{j:\hat{x}_j=1} x_j + \sum_{j:\hat{x}_j=0} (1 - x_j) \le n^x - 1, \tag{3}$$

where x_j is the jth element of binary vector variable x and n^x is the dimension of binary vector x. Note that only a finite number of cuts (3) can be generated.

Algorithm 1 Benders Decomposition Method

1: Initialize $X_0^0 \leftarrow X_0$ and $k \leftarrow 0$.
2: Find $(x^k, y_0^k) \in \arg\min_{(x,y_0)\in X_0^k} \{c^T x + d^T y_0\}$.
3: **while** $\exists s \in \mathcal{S}^C$ such that $\{y : (x^k, y) \in X_s\} = \emptyset$ **do**
4: Update $X_0^{k+1} \leftarrow X_0^k \cap \{x : (3) \text{ generated at } x^k\}$.
5: Set $k \leftarrow k + 1$.
6: Find $(x^k, y_0^k) \in \arg\min_{(x,y_0)\in X_0^k} \{c^T x + d^T y_0\}$.
7: **end while**

The Benders decomposition method is summarized in Algorithm 1. The algorithm initializes the set of feasible solutions for the non-contingency scenario (line 1). The Benders master problem is solved to find a candidate solution (lines 2 and 6). If the solution is violated by a contingency scenario, it is excluded from the solution set by adding the feasibility cut (3) (lines 3 and 4). Note that the feasibility check in line 3 can be parallelized for each scenario. Since there exists a finite number of binary variable values in the first stage, the algorithm terminates after generating a finite number of feasibility cuts.

III. Parallel Dual Decomposition Methods

In this section, we present parallel dual decomposition (DD) methods for solving the Lagrangian dual of SMIP by relaxing the nonanticipativity constraints.

A. Dual Decomposition

We consider the simplified formulation of model (1) as

$$\min_{x_s, y_s} \sum_{s\in\mathcal{S}} \pi_s \left(c^T x_s + d^T y_s\right) \tag{4a}$$

$$\text{s.t.} \sum_{s\in\mathcal{S}} H_s x_s = 0, \tag{4b}$$

$$(x_s, y_s) \in X_s,\ s \in \mathcal{S}, \tag{4c}$$

where (4b) represents the nonanticipativity constraints (1b) and (1c) and set X_s is the set of feasible solutions (x_s, y_s) that represents all the other constraints (1d) – (1r) for each scenario. Note also that both x_s and y_s are mixed-binary variables.

DD is obtained by the Lagrangian relaxation of the nonanticipativity constraints (4b) to the objective function. The goal of this approach is to find the best Lagrangian dual bound by solving

$$z_{LD} := \max_{\lambda} \sum_{s\in\mathcal{S}} D_s(\lambda), \tag{5}$$

where for $s \in \mathcal{S}$ the Lagrangian dual function is defined as

$$D_s(\lambda) := \min_{(x,y)\in X_s} \pi_s \left(c^T x + d^T y\right) - \lambda^T H_s x \tag{6}$$

and λ is the Lagrangian multiplier for the nonanticipativity constraints (4b). We call problem (6) the Lagrangian subproblem. We also define $D(\lambda) := \sum_{s\in\mathcal{S}} D_s(\lambda)$.

B. Asynchronous Bundle Method with Trust Region

We present a bundle method for solving the Lagrangian dual problem (5). In particular, a TR constraint is added in order to regularize the search space for the dual variable λ. The bundle method solves problem (5) by approximating the Lagrangian dual function $D_s(\lambda)$ with a set of linear inequalities. Let k and l be the index of major and minor iterations, respectively. The major iteration updates the best dual bound, whereas the minor iteration approximates the Lagrangian dual problem.

We first define the model function

$$\tilde{m}_{k,l}(\lambda) := \max_{\theta_s} \sum_{s\in\mathcal{S}} \theta_s \tag{7a}$$

$$\text{s.t. } \theta_s \le D_s(\lambda^i) + (H_s x_s^i)^T (\lambda - \lambda^i),$$
$$i \in \mathcal{B}^{k,l},\ s \in \mathcal{S}^i, \tag{7b}$$

where linear inequalities (7b) outer-approximate the Lagrangian dual function $D_s(\lambda)$, $\mathcal{B}^{k,l}$ is the set of indices for the linear inequalities at iteration (k,l), and $\mathcal{S}^i \subseteq \mathcal{S}$ represents the subset of scenario indices such that the linear inequalities are generated at λ^i. A subgradient of D_s at λ^i is given by $H_s x_s^i$, where x_s^i is an optimal solution to the Lagrangian subproblem for each $s \in \mathcal{S}$. Moreover, we update the model function asynchronously by using only a subset of the Lagrangian subproblems, while the other subproblems are being solved.

Let Π be the set of processes, each $\pi \in \Pi$ of which solves a subset $\mathcal{S}_\pi \subseteq \mathcal{S}$ of the scenario subproblems. We denote the set of processes available for subproblem solutions at iteration (k,l) by $\Pi_{k,l} \subseteq \Pi$. Let $\underline{\Pi}$ be the minimum number of processes, from which the master problem receives the subproblem solutions. We define a queue of dual variables available for evaluation at iteration (k,l), denoted by $\Lambda_{k,l}$. Let $\bar{\Lambda}$ be the maximum capacity of the queue. In addition, we let $\Lambda_{k,l}^R(\pi), \Lambda_{k,l}^C(\pi) \subseteq \Lambda_{k,l}$ be the queues containing the dual variables *ready* and *complete*, respectively, for evaluation at process π.

At iteration (k,l) the master problem is given by

$$\max_{\lambda} \left\{\tilde{m}_{k,l}(\lambda) : \|\lambda - \lambda^k\|_\infty \le \Delta_{k,l}\right\}, \tag{8}$$

where λ^k is the TR center and $\Delta_{k,l}$ is the TR radius. Updating the TR constraint in (8) is important for the convergence of the bundle method. We perform a descent test to update the TR center. In particular, if

$$D(\hat{\lambda}^{k,l}) \ge D(\lambda^k) + \xi[\tilde{m}_{k,l}(\lambda^{k,l}) - D(\lambda^k)], \tag{9}$$

we update the TR center by the current iterate $\lambda^{k,l}$ and the best dual bound, where $\hat{\lambda}^{k,l} \in \cap_{\pi\in\Pi}\Lambda_{k,l}^C(\pi)$ is the dual variable evaluated by all processes $\pi \in \Pi$. Such a step is called a *serious step*. If the model function $m_{k,l}$ is updated instead, the step is called a *null step*.

The asynchronous algorithm is summarized in Algorithm 2. The algorithm is initialized with an initial trial point λ^0 and parameters $\epsilon \ge 0, \Delta_{0,0} \in (0, \bar{\Delta}], \xi \in (0, 0.5), \bar{\Lambda} > 0, \underline{\Pi} > 0, J_\pi \subseteq J$ for $\pi \in \Pi$. Note that Algorithm 2 becomes the synchronous counterpart when $\underline{\Pi} = |\Pi|$. For the initial trial

Algorithm 2 Asynchronous Bundle-Trust-Region Method

1: Initialize λ^0 and parameters, and set $k \leftarrow 0$ and $l \leftarrow 0$.
2: Evaluate $D_s(\lambda^0)$ for $s \in \mathcal{S}$.
3: Initialize the model function $\tilde{m}_{0,0}$.
4: **loop**
5: Solve the master (8) to find $\lambda^{k,l}$.
6: **if** $\tilde{m}_{k,l}(\lambda^{k,l}) - D(\lambda^k) \le \epsilon(1+|D(\lambda^k)|)$ **then**
7: Stop
8: **end if**
9: **if** $|\Lambda_{k,l}| < \bar{\Lambda}$ **then**
10: $\Lambda^R_{k,l}(\pi) \leftarrow \Lambda^R_{k,l}(\pi) \cup \{\lambda^{k,l}\}$ for each $\pi \in \Pi$.
11: **end if**
12: **for** $\pi \in \Pi_{k,l}$ **do**
13: **if** $\Lambda^R_{k,l}(\pi) = \emptyset$ **then**
14: SEND $\lambda^{k,l}$ to process π.
15: **else**
16: Choose the first element $\lambda^q \in \Lambda^R_{k,l}(\pi)$.
17: SEND λ^q to process π.
18: **end if**
19: $\Pi_{k,l} \leftarrow \Pi_{k,l} \backslash \{\pi\}$.
20: **end for**
21: **repeat**
22: RECEIVE $D_j(\lambda^q)$ and x^q_j for $j \in J_\pi$ from π.
23: (Evaluate x^q_j for upper bounds.)
24: $\Pi_{k,l} \leftarrow \Pi_{k,l} \cup \{\pi\}$.
25: **until** $|\Pi_{k,l}| \ge \underline{\Pi}$
26: `serious` $\leftarrow$ `false`.
27: **if** $\cap_{\pi\in\Pi}\Lambda^C_{k,l}(\pi) \neq \emptyset$ **then**
28: $\hat{\lambda}^{k,l} \leftarrow \arg\max\{D(\lambda) : \lambda \in \cap_{\pi\in\Pi}\Lambda^C_{k,l}(\pi)\}$
29: $\Lambda_{k,l} \leftarrow \Lambda_{k,l}\backslash\{\hat{\lambda}^{k,l}\}$
30: **if** descent test (9) holds **then**
31: Choose $\Delta_{k+1,0} \in [\Delta_{k,l}, \Delta^{max}]$.
32: Choose $\Lambda_{k+1,0} \subseteq \Lambda_{k,l}$.
33: Set $\lambda^{k+1} \leftarrow \hat{\lambda}^{k,l}, \tilde{m}_{k+1,0} \leftarrow \tilde{m}_{k,l}, \Pi_{k+1,0} \leftarrow \Pi_{k,l}, k \leftarrow k+1$ and $l \leftarrow 0$.
34: `serious` $\leftarrow$ `true`.
35: **else**
36: Choose $\Delta_{k,l+1} \in (0, \Delta_{k,l}]$.
37: **end if**
38: **end if**
39: **if** `serious` $=$ `false` **then**
40: Update $\tilde{m}_{k,l+1}$ by adding cuts (7b).
41: Set $\Pi_{k,l+1} \leftarrow \Pi_{k,l}, \Lambda_{k,l+1} \leftarrow \Lambda_{k,l}, l \leftarrow l+1$.
42: **end if**
43: **end loop**

point λ^0, the Lagrangian subproblems are solved in parallel (line 2) and synchronized to initialize the model function with the initial bundle information (line 3). The master problem is solved to find a new trial point in line 5. The algorithm terminates (line 7) if the model function value is close to the best dual bound $D(\lambda^k)$ within the gap of $\epsilon(1+|D(\lambda^k)|)$ in line 6. If the queue is not full, the new trial point is stored to the queue and ready for evaluation (lines 9–11). In lines 12–24, the master process asynchronously sends and receives necessary data with some subproblem processes. The master process sends each available process either the new trial point (line 14) or the first element of the dual variables ready for evaluation (line 16–17). Then, the master receives bundle information from at least $\underline{\Pi}$ processes (lines 21–24). The primal solution x^q_j may be evaluated to obtain upper bounds, which can also be done on separate processes in parallel. The descent test (9) is performed only for the dual variable evaluated by all the scenarios (lines 26–37). Otherwise, the model function is updated (lines 38–41). For $\epsilon > 0$, Algorithm 2 terminates after a finite number of steps, as shown in [8].

IV. Computational Study

In this section, we present the numerical results by using large-scale test problem instances of the SCUC model. We implemented the Benders decomposition (Algorithm 1) in `Julia` [9] by using `CPLEX` callback function and MPI library. For the DD methods (Algorithm 2 and the synchronous counterpart) we use a parallel decomposition solver `DSP` [7], which can read an optimization model from `Julia` and run on a high-performance computing clusters by using the MPI library. The master problem and Lagrangian scenario subproblems were solved by using `CPLEX-12.7` with the relative gap tolerance of 10^{-5}. In particular, the master problem was solved by the barrier optimizer in `CPLEX`. We have modeled the CCUC problem (1) with the `JuMP` modeling package [10] in `Julia`. All experiments were run on the 1024-node computing cluster *Bebop* at Argonne National Laboratory.

A. Problem Instances

We use the test data that represents the California Independent System Operator system interconnected to the Western Electricity Coordinating Council (WECC), as also used in [2]. The WECC test system consists of 225 buses, 375 transmission lines, 130 generation units, 40 loads, 5 import points, 5 wind farms, and 11 non-wind renewable generation units. We set 90 of the 130 generators to be capable of starting in response to a contingency event (i.e., *fast* generators). We consider a 24-hour time horizon with hourly time intervals. As a result, each problem instance has 1,000 binary variables, one continuous variable, and no constraint in the first stage and 8,750 binary variables, 15,734 continuous variables, and 35,149 constraints in the second stage for each scenario.

Eight problem instances were generated for each of spring, summer, fall, and winter and for each weekday (WD) and weekend (WE). The total net load for each time $t \in \mathcal{T}$ is shown in Figure 1 for each problem instance. Moreover, as modeled in (1), we allow predetermined amount of load lost at a penalty cost $\rho = 1,000 \sum_{j\in\mathcal{D}} \sum_{t\in\mathcal{T}} D_{jt}$ \$ per fraction ϵ_s, which is equivalent to \$1,000 per MWh of load lost.

We consider each contingency event as a single transmission line failure. However, we assume that 6 of the 375 lines are protected and invulnerable, as removing each of the 6 lines creates an island of the system. This results in 370 contingency scenarios. In addition, we consider three wind power

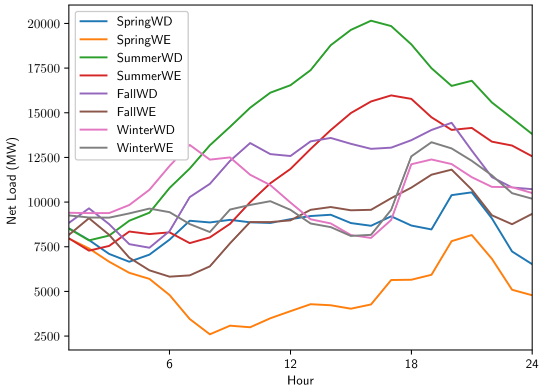


Figure 1. Total net load of the WECC 225-bus test system for each instance

TABLE I
COMPUTATIONAL RESULTS OF THE BENDERS DECOMPOSITION METHOD

Instance	Objective	Active Contingencies	Time
SpringWD	1654734	L321, L326, L352, L353	464
SpringWE	1143749	L326, L352, L353	1151
SummerWD	4890148	L216, L296, L321, L322 L326, L352, L353	630
SummerWE	3300184	L321, L326, L352, L353	542
FallWD	2492479	L321, L326, L352, L353	631
FallWE	1589561	L321, L326, L352, L353	540
WinterWD	1947431	L321, L326, L352, L353	559
WinterWE	1346567	L321, L326, L352, L353	435

generation scenarios of mild, medium, and high penetration levels with 43,162, 76,290, and 100,169 MW, respectively. We assume that each scenario has equal probability.

B. *Deterministic Wind Power Generation*

We present the computational results from the Benders decomposition (Algorithm 1) for solving the SCUC model with the 370 contingency scenarios for a given wind power generation, as described in Section II-C. In particular, we aim to find a list of contingency scenarios (i.e., *active* contingency scenarios) that are used to generate feasibility cuts (3) in the Benders decomposition. We solve eight problem instances for each wind scenario and report the mean objective values for the eight problem instances. Each problem instance was solved in parallel with 371 computing cores of the cluster.

Table I reports mean objective value, active contingency events, and mean time in second for each problem instances. The active contingency events are the transmission lines, representing the scenarios used to generate at least one feasibility cut in Algorithm 1. The contingency scenarios on lines L326, L352, and L353 are commonly active for all the eight problem instances, and the contingency scenario of line L321 are also active for six instances. For the SummerWD instance, additional scenarios on lines L216 and L296 were used to generate the feasibility cuts. The average solution times for the SCUC model (with 370 contingency scenarios and one wind power scenario) range from 435 to 1,151 seconds in wall-clock time.

TABLE II
COMPUTATIONAL RESULTS AND LOAD IMBALANCES OF THE SYNCHRONOUS DD METHOD

Instance	UB	LB	Gap	Iter	$\bar{\nu}$ (ν^{min} – ν^{max})
SpringWD	3991	3398	15%	761	104 (43 – 461)
SpringWE	2741	2667	3%	910	65 (20 – 319)
SummerWD	35296	7103	80%	865	67 (27 – 339)
SummerWE	18513	4448	76%	993	80 (34 – 584)
FallWD	5622	4471	20%	799	129 (68 – 421)
FallWE	3538	3425	3%	847	52 (28 – 366)
WinterWD	4945	3147	36%	731	107 (63 – 357)
WinterWE	3288	2980	9%	829	67 (33 – 322)

C. *Synchronous Dual Decomposition*

We present the computational results from the synchronous DD for the problem instances. For the numerical experiments of the DD methods, we use the combination of the wind power generation scenarios and the active contingency scenarios L321, L326, L352, and L353, as reported in Section IV-B, resulting in 15 scenarios for each problem instance. Each problem instance was solved by using 36 cores of the cluster with 10 hours of time limit. 15 cores were used for the scenario subproblems, another 15 cores were for upper bounding, and the other cores were used for solving the master.

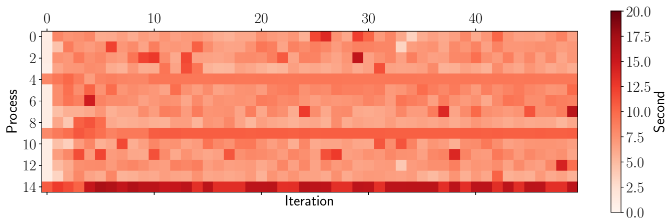
We use the percent imbalance metric, as defined in [11], for quantifying the load imbalance of solving the Lagrangian subproblems. The percent imbalance metric ν_k for each iteration k is given by $\nu_k := (t_k^{max}/\bar{t}_k) \times 100\%$, where t_k^{max} and $\bar{t}_k$ are the maximum and mean subproblem solution times, respectively, for each iteration k.

Table II reports the computational results from the synchronous counterpart of Algorithm 2 by setting $\underline{\Pi} = |\Pi|$. for solving each problem instance. None of the problem instances found optimal Lagrangian dual bounds in the 10-hour time limit. The best upper and lower bounds ($\times 10^3$) and duality gaps are reported in the table. The maximum, minimum, and mean percent imbalance metrics are also reported in columns ν^{max}, ν^{min}, and $\bar{\nu}$, respectively. For all the problem instances, every iteration suffers from the load imbalance of at least 20%. The maximum imbalance was 584% for the SummerWE instance.

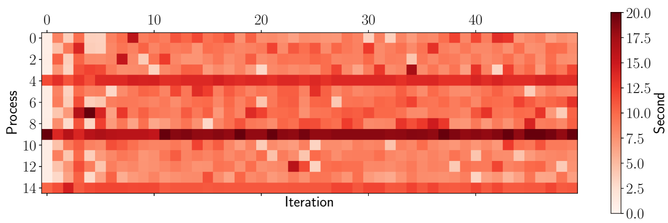
Figure 2 is a heatmap showing the Lagrangian subproblem solution time for each iteration and for each worker process, which visualizes the seriousness of load imbalances among the processes. For the highly imbalanced SummerWE instance in Figure 2b, processes 4, 9, and 14 were almost always significant bottlenecks of the DD. These processes were assigned to solve the scenario subproblems with nonzero probabilities, of which the objective functions are more complicated than those of the other scenario subproblems.

D. *Asynchronous Dual Decomposition*

We present the computational performance of the asynchronous DD (Algorithm 2) for the same problem instances, as used in Section IV-C. Table III reports the computational results and relative speedup for each instance. The speedup was calculated by 10 hours (of the time limit) divided by the



(a) SpringWE (Marginally Imbalanced)



(b) SummerWE (Highly Imbalanced)

Figure 2. Load imbalance of solving the Lagrangian subproblems for two different problem instances (SpringWE and SummerWE). This figure plots the first 50 iterations only for visibility.

TABLE III
COMPUTATIONAL RESULTS AND RELATIVE SPEEDUP OF THE ASYNCHRONOUS DD METHOD

Instance	UB	LB	Gap	Iter	Time	Speedup
SpringWD	3695	3564	4%	2033	20205	3.60
SpringWE	2985	2669	10%	1858	13102	3.12
SummerWD	29307	25494	13%	3498	36000	8.08
SummerWE	10397	7905	24%	2499	23349*	7.05
FallWD	5378	4956	8%	2346	36000	4.77
FallWE	3559	3479	2%	2348	19168	3.60
WinterWD	4068	3988	2%	2162	16586	7.12
WinterWE	3182	3065	4%	2167	22593	3.38

* terminated due to numerical issue in CPLEX

asynchronous solution time at the iteration that found the best lower bound larger than that reported in Table II. All the instances, except SummerWD, SummerWE and FallWD, found optimal Lagrangian bounds. The duality gaps are significantly lower than those found in Table II. We note that the number of iterations in the asynchronous method is significantly larger than those of the synchronous version, because the master problem is solved with only a subset of the subproblem solutions (while slow subproblems are being solved). The asynchronous solution resulted in larger duality gap for Spring WE than did the synchronous, because the upper bounds were only heuristically found. The ability to solve the master problem with fewer scenarios, however, allows the asynchronous algorithm to make progress while slow processes terminate. Consequently, the asynchronous solution times are reduced for most problem instances by up to a factor of 8.08. The lowest speedup achieved is 3.12 for the SpringWE instance, which is a marginally imbalanced instance as shown in Figure 2a.

V. Conclusions and Future Work

We presented a parallel, asynchronous DD method in application to solving SCUC model with contingency events on system components in combination with uncertain (wind power) generation forecast. Adopting the SMIP formulation of the SCUC model used in [2], we also introduce a slack variable to avoid any infeasible problem instances due to the problem data, representing a predetermined amount of total system load lost, as used in [4]. We also presented a special case of the SCUC model that considers a deterministic power generation with contingency scenarios, which allows us to develop a parallel Benders decomposition. In particular, this model was used to find *active* contingency scenarios for feasibility of the SCUC model solution.

We applied a recently-developed asynchronous DD method for efficiently solving the problem in parallel. In the computational experiments using the WECC 225-bus test system, we found that the asynchronous method can significantly accelerate the solution time and also find tighter upper and lower bounds, as compared with the synchronous counterpart. In addition, we found that the SCUC model is much more difficult (i.e., it requires more iterations and wall clock time) than the stochastic unit commitment with uncertain wind generation (see [8] using the same problem instances). In particular, the latter has random right-hand sides only, whereas the former has both random matrices and right-hand sides.

Future work of interest includes the development of effective algorithms for solving SMIP problems with both random matrices and right-hand sides, particularly aiming to reduce the number of iterations of the DD method.

References

[1] L. Wu, M. Shahidehpour, and T. Li, "Stochastic security-constrained unit commitment," *IEEE Transactions on Power Systems*, vol. 22, no. 2, pp. 800–811, 2007.

[2] A. Papavasiliou and S. S. Oren, "A comparative study of stochastic unit commitment and security-constrained unit commitment using high performance computing," in *Control Conference (ECC), 2013 European*. IEEE, 2013, pp. 2507–2512.

[3] Q. Wang, J.-P. Watson, and Y. Guan, "Two-stage robust optimization for n–k contingency-constrained unit commitment," *IEEE Transactions on Power Systems*, vol. 28, no. 3, pp. 2366–2375, 2013.

[4] R. L.-Y. Chen, N. Fan, A. Pinar, and J.-P. Watson, "Contingency-constrained unit commitment with post-contingency corrective recourse," *Annals of Operations Research*, vol. 249, no. 1-2, pp. 381–407, 2017.

[5] C. C. CarøE and R. Schultz, "Dual decomposition in stochastic integer programming," *Operations Research Letters*, vol. 24, no. 1, pp. 37–45, 1999.

[6] A. Papavasiliou, S. S. Oren, and R. P. O'Neill, "Reserve requirements for wind power integration: A scenario-based stochastic programming framework," *IEEE Transactions on Power Systems*, vol. 26, no. 4, pp. 2197–2206, 2011.

[7] K. Kim and V. M. Zavala, "Algorithmic innovations and software for the dual decomposition method applied to stochastic mixed-integer programs," *Mathematical Programming Computation (to appear)*, 2017.

[8] K. Kim, C. G. Petra, and V. M. Zavala, "An asynchronous bundle-trust-region method for dual decomposition of stochastic mixed-integer programming," Argonne National Laboratory, Tech. Rep. ANL/MCS-8046-0917, 2017.

[9] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM Review*, vol. 59, no. 1, pp. 65–98, 2017.

[10] I. Dunning, J. Huchette, and M. Lubin, "Jump: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[11] O. Pearce, T. Gamblin, B. R. De Supinski, M. Schulz, and N. M. Amato, "Quantifying the effectiveness of load balance algorithms," in *Proceedings of the 26th ACM international conference on Supercomputing*. ACM, 2012, pp. 185–194.